Research Project

EFFICIENTLY DETERMINING

THE SHORTEST SERVICE ROUTES

Can DENIZCI

IZMIR
October - 2023

CONTENTS

ABSTRACT

This work considers the Vehicle Routing Problem, which has several real-life applications. The Vehicle Routing Problem is an optimization problem that tries to minimize the travel distance traveled by vehicles with one or more warehouses and a certain number of customers or cities while distributing products. Companies need to minimize such a cost in today's world, where fuel prices constantly increase.

In this project, an algorithm was developed to solve the Vehicle Routing Problem, and a computer program was coded in Python. With the help of the developed program, the service routes of a private company were determined. The computational results show that the founded routes by the developed program are shorter than the existing routes followed by the services.

Keywords: Vehicle Routing Problem, Optimization, Algorithms, Nearest-Neighbor.

1. INTRODUCTION

Transportation planning involves a strategic process where current and future transportation needs are assessed, and comprehensive policies and goals are formulated to effectively address those needs. This process entails an analysis of existing transportation infrastructure, assets, and resources to develop a long-term vision for a sustainable and efficient transportation system. The optimization of transportation planning is crucial for improving the efficiency of transportation systems and bridging the gap between long-term strategic objectives and short-term operational excellence. This process is responsible for the implementation and realization of the foundational strategic goals established by transportation planning. While transportation planning encompasses strategic decision-making, such as the identification of requirements, infrastructure development, policy formulation, and resource allocation, optimized transportation planning ensures the effective execution of these decisions in day-to-day operations. By continually optimizing routes, schedules, and resource allocation, transportation systems can swiftly adapt to changing conditions.

Routing problems are a class of optimization problems that arise in various fields, including operations research, transportation engineering, and logistics. These problems interest finding the most efficient path or set of paths for pedestrians/ vehicles to traverse from a source to a destination while satisfying certain constraints or objectives. The fundamental goal of routing problems is to determine optimal or near-optimal routes, considering factors such as distance, cost, time, or other relevant criteria. The optimization criteria may vary based on the specific context of the problem. The Traveling Salesman Problem (TSP), which is one of the famous routing problems, seeks to find the shortest possible tour that visits a set of given cities and returns to the starting city. The Multiple Traveling Salesman Problem (mTSP) is a natural generalization of the well-known TSP, involving multiple salesmen visiting a given number of cities exactly once and returning to the initial position with the minimum traveling cost. The mTSP closely aligns with the overarching goals of transportation planning by highlighting the imperative of determining the most efficient routes among multiple destinations. An important extension of mTSP that is more suitable for real-life applications is the Vehicle Routing Problem (VRP). The VRP represents a challenging combinatorial optimization problem that aims to determine a route plan for a fleet of vehicles sharing uniform capacity, a central (or multiple) depot, and numerous customer requests while striving to minimize the overall route cost, typically measured in terms of the distance covered by the vehicles. The VRP serves as a generalization of the TSP, and various algorithms based on the mTSP can also be applied to multiple VRPs by incorporating additional constraints, particularly the relaxation of capacity restrictions. The VRP finds widespread application in real-life

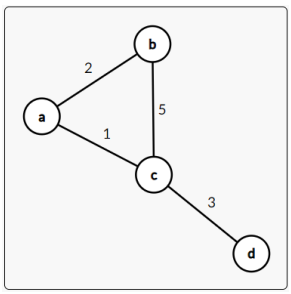
scenarios, containing fields such as robotics, transportation, and networking. In this context, ongoing endeavors are directed toward the optimization of routes, schedules, and resource allocation to ensure swift adaptation to changing conditions.

The aim of this project is to develop an effective plan for a private company's employee transportation services. For this purpose, an algorithm based on Nearest Neighor is proposed. The project seeks to achieve efficient routes via the proposed algorithm and reduce the total travelling distance of the vehicles.

This project is structured as follows: Section II presents the addressed problem, and a literature review is given in Section III. The IV section introduces the proposed algorithm for the problem. In Section V, the application and results are presented. Lastly, the concluding section provides a general evaluation of the study and offers some recommendations for future research.

2. THE ADRESSED PROBLEM

Graph theory, a fundamental branch of mathematics, provides an adaptable framework for modeling and solving complex problems in various domains. Graph theory has emerged as a powerful tool for analyzing and optimizing problems related to network structures. By representing transportation networks as graphs, theory algorithms can contribute to the efficient solution of VRP, a crucial problem in logistics and transportation management. In graph theory, a graph *G* is defined as a pair *(V,E)*, where *V* represents a set of vertices (nodes) and *E* represents a set of edges connecting the vertices. Edges may have weights to denote distances, costs, or time between nodes. A distance matrix is a square matrix that represents the distances between pairs of vertices in a graph. Figure 1 gives an example of a simple graph and its distance matrix.



	a	b	c	d
a	0	2	1	0
b	2	0	5	0
c	1	5	0	3
d	0	0	3	0

Figure 1. A simple graph and its distance matrix.

VRP is a combinatorial optimization problem that addresses the efficient allocation of a fleet of vehicles to serve a set of customers, minimizing the total travel distance or time while satisfying various constraints. The problem is NP-hard, therefore to solve ARP requires effective algorithms. In the addressed problem, *m* vehicle starts from different location, pick up the employees and take them to the company where each vehicle has a passenger capacity. Thus, the addressed problem is a Single Depot Open Vehicle Routing Problem (SD_OVHP) where the vehicles do not return to the starting point after delivering the employees to the company. Figure 2 represents the problem on the map.

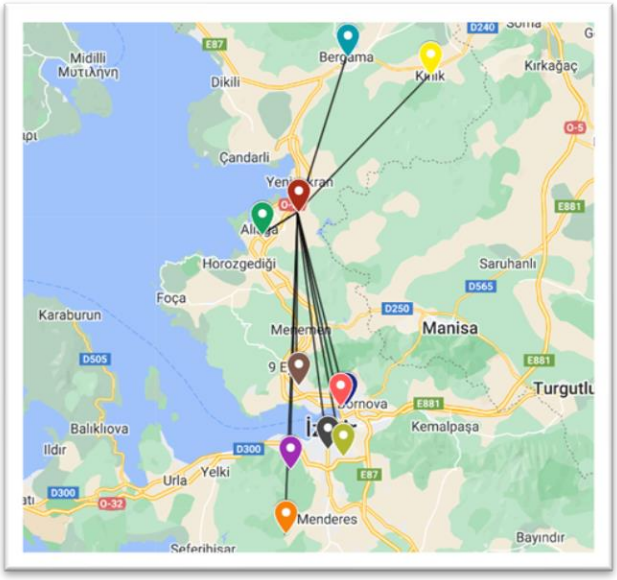


Figure 2. A representative route to the company from various starting points is depicted on the map.

In this problem, each vehicle and employee are represented as node in the graph. Edge weights in the graph correspond to the distances between vehicle/employed locations.

3. LITERATURE REVIEW

This section presents some studies on VHP, arranged in chronological order.

Schneider et al. (2015) studied the Vehicle Routing Problem with intermediate stops where vehicles have to stop at certain locations along their routes to continue their service. At these stops, the vehicles replenish or unload their cargo or they stop to refuel. The authors proposed an adaptive variable neighborhood search to solve the considered problem.

Mohammed et al. (2017) aimed to find the shortest route for the Vehicle Routing Problem to help UNITEN University reduce student's transportation costs by using. genetic algorithm.

Zhang et al. (2017) considered the Vehicle Routing Problem with time window and pallet loading constraints. The considered problem accounts for the actual needs of businesses in the logistics industry such as the delivery of consumer goods and agricultural products. The authors proposed we a hybrid approach by combining Tabu Search and the Artificial Bee Colony algorithm to solve the considered problem.

Sajid et al. (2021) studied the capacitated vehicle routing problem and proposed a novel crossover operator for evolutionary algorithms, which works stochastically to search for the optimal solutions.

Huang et al. (2022) addressed the Vehicle Routing Problem with Drone (VRPD) by assigning customers to drone-truck pairs, determining the number of dispatching drone-truck units, and travel costs of both vehicles are minimized. To solve this problem, the authors used Ant Colony Optimization.

For a more comprehensive survey of VRP, "Vehicle Routing Problem and Related Algorithms for Logistics Distribution: A Literature Review and Classification" by Konstantakopoulos et al. (2020) can be examined.

4. THE PROPOSED ALGORITHM

In the addressed problem, *m* vehicles start from different locations, pick up the employees, and take them to the company where each vehicle has a passenger capacity. The proposed algorithm aims to minimize the total cost of service routes. The algorithm is based on the Nearest Neighbor algorithm, which is one of the well-known algorithms for solving TSP (Reinelt (2023)). The Nearest Neighbor algorithm is a simple and intuitive heuristic method where the goal is to find the shortest possible tour that visits each city exactly once and returns to the starting city. The Nearest Neighbor algorithm works as follows: Start at a given city (the starting point), select the nearest unvisited city, and move to the selected city. Then, mark the selected city as visited and repeat these steps until all cities are visited. Lastly, return to the starting city to complete the tour.

Similar to the Nearest Neighbor algorithm, the proposed algorithm constructs *m* tours incrementally by always choosing the nearest unvisited city at each step. Unlike the TSP, in the addressed problem *m* vehicles starts from different locations. Thus, in the proposed

algorithm the nearest unvisited point, which can be reached from *m* different starting points, is selected. Then, the locations of the vehicles (starting points) are updated, and the previous step is repeated until all points are visited. The main steps of the proposed algorithm are given in Figure 3.

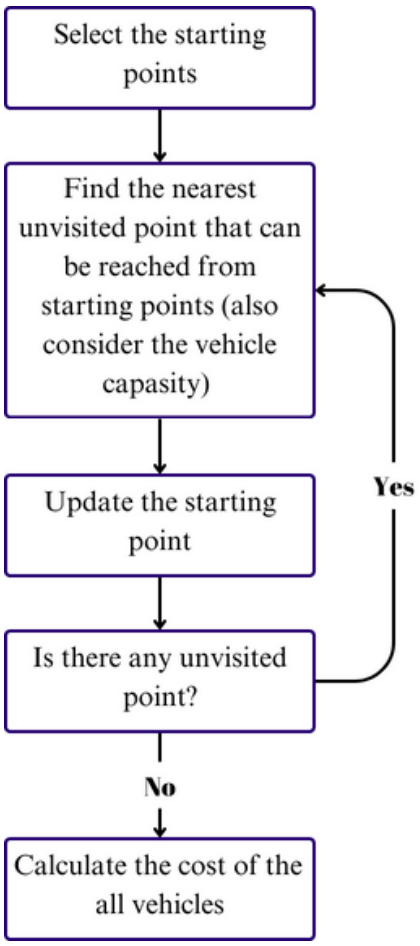


Figure 3. The main steps of the proposed algorithm.

4.1 The Computer Program

The proposed algorithm is executed using the Python 3.9.7 programming language. The program accepts a distance matrix to find service routes. Thus, a text document (.txt) file is used to store the distance matrix, which comprises the distances between each location. In addition, the last line of the text document represents the number of the people in each location.

In the developed program, a cost matrix (graph[][]) is used to represent the travel cost between each pair of stops. It begins by initializing various arrays and matrices, including the

routes matrix (routes[][]) that represents the routes taken by each vehicle. The developed program starts with determining the starting points of all vehicles. In my application, I determined the starting points manually (Figure 4).

```
def starting_points():

    routes[0][0] = 0
    number_of_pinvh[0] += number_of_people[0]
    routes[1][0] = 3
    number_of_pinvh[1] += number_of_people[3]
    routes[2][0] = 16
    number_of_pinvh[2] += number_of_people[16]
    routes[3][0] = 19
    number_of_pinvh[3] += number_of_people[19]
    routes[4][0] = 28
    number_of_pinvh[4] += number_of_people[28]
    routes[5][0] = 35
    number_of_pinvh[5] += number_of_people[35]
    routes[6][0] = 46
    number_of_pinvh[6] += number_of_people[46]
    routes[7][0] = 56
    number_of_pinvh[7] += number_of_people[56]
    routes[8][0] = 67
    number_of_pinvh[8] += number_of_people[67]
    routes[9][0] = 7
    number_of_pinvh[9] += number_of_people[7]

    for i in range(10):
        r_indis[i] = 0

    visited[0] = 1
    visited[3] = 1
    visited[16] = 1
    visited[19] = 1
    visited[28] = 1
    visited[35] = 1
    visited[46] = 1
    visited[56] = 1
    visited[67] = 1
    visited[7] = 1
    visited[72] = 1
```

Figure 4. Manually determining the starting points of the vehicles.

After the starting points are initialized for each vehicle, the program then enters an iterative loop for each vehicle to select the next location to be visited (Figure 5). It selects the nearest unvisited location (visited[]==0) that not exceeding the capacity of the service (Note that, in any location, more than one person can be taken by the services - number_of_people[]). The algorithm then updates the route matrix, and the number of people (number_of_pinvh[]) of the selected vehicle. The loop continues until all locations have been visited.

```
stop = True
while stop:

    minimum = 10000
    for i in range(n):
        if visited[i] == 0:
            for j in range(m):
                if graph[int(routes[j][int(r_indis[j])])][i] < minimum and number_of_pinvh[j] + number_of_people[i] < cap:
                    minimum = graph[int(routes[j][int(r_indis[j])])][i]
                    next_point = i
                    vh = j

    r_indis[vh] = r_indis[vh] + 1
    routes[vh][int(r_indis[vh])] = next_point
    visited[next_point] = 1
    number_of_pinvh[vh] = number_of_pinvh[vh] + number_of_people[next_point]

    temp = 0
    for i in range(n):
        if visited[i] == 1:
            temp += 1

    if temp == n:
        stop = False
```

Figure 5. Manually determining the starting points of the vehicles.

Note that, the program starts by setting the visited value of the target point (the location of the company) to 1. After all points are visited and all the routes are determined, the target point is added to the end of all routes (Figure 6).

```
for i in range(m):
    r_indis[i] = r_indis[i] + 1
    routes[i][int(r_indis[i])] = 72
```

Figure 6. Adding the target point to the end of all routes.

Lastly, after determining all routes, the cost of each route is calculated by summing the costs of the edges in the route (Figure 7).

```
totalcost = 0
for i in range(m):
    cost[i] = 0
    for j in range(int(r_indis[i])):
        cost[i] = cost[i] + graph[int(routes[i][j])][int(routes[i][j+1])]
    totalcost = totalcost + cost[i]
```

Figure 7. Calculating the distances.

5. THE CASE STUDY

The aim of this study is to find near optimal routes for the services of a private company. Thus, I applied the developed program on real-life data and compared the results with the current situation.

5.1 The Dataset

The dataset consists of the addresses of the initial positions of services and stops used by employees of a private company. In my application, there are 10 services and 73 stop positions (including the location of the company). Due to privacy concerns, this specific dataset will not be disclosed. Initially, the dataset is converted into a distance matrix for the developed program algorithm.

```python
import math

def haversine_distance(coord1, coord2):
    # Dünyanın yarıçapı (km bazında)
    R = 6371.0
   #R = 3958.8 mil

    # İki nokta arasındaki koordinat farklarını radyan cinsinden hesaplama
    lat1, lon1 = math.radians(coord1[0]), math.radians(coord1[1])
    lat2, lon2 = math.radians(coord2[0]), math.radians(coord2[1])

    # Haversine formülü ile kullanarak iki nokta arasındaki mesafeyinin hesaplanması
    dlat = lat2 - lat1
    dlon = lon2 - lon1
    a = math.sin(dlat / 2)**2 + math.cos(lat1) * math.cos(lat2) * math.sin(dlon / 2)**2
    c = 2 * math.atan2(math.sqrt(a), math.sqrt(1 - a))
    distance = R * c

    return distance

```

Figure 8. Calculating the Haversine distances between locations.

To achieve this, the coordinates of each stop location were obtained from Google Maps. Utilizing these coordinates, the Haversine distance formula was employed to calculate the distances between each stop location (Figure 8). The Haversine distance formula is as follows:

$$d = 2 \cdot R \cdot arcsin\left(\sqrt{\sin^2\left(\frac{\phi_2 - \phi_1}{2}\right) + cos(\phi_1) \cdot cos(\phi_2) \cdot \sin^2\left(\frac{\lambda_2 - \lambda_1}{2}\right)}\right)$$

where ϕ_1, ϕ_2 are the latitude of point 1 and latitude of point 2, and λ_1, λ_2 are the longitude of point 1 and longitude of point 2 (Korn et al. (2000)).

Subsequently, a distance matrix was constructed (Figure 9), considering the results obtained.

```python
def calculate_distance_matrix(coords):
    num_points = len(coords)
    distance_matrix = [[0] * num_points for _ in range(num_points)]

    for i in range(num_points):
        for j in range(num_points):
            distance_matrix[i][j] = haversine_distance(coords[i], coords[j])

    return distance_matrix
```

Figure 9. Calculating the Distance Matrix.

5.2 The Results

To test the efficiency of the developed program, I applied the developed program on real-life data and compared the results with the current situation. Figure 10 gives total distance to be traveled for each service in kilometer.

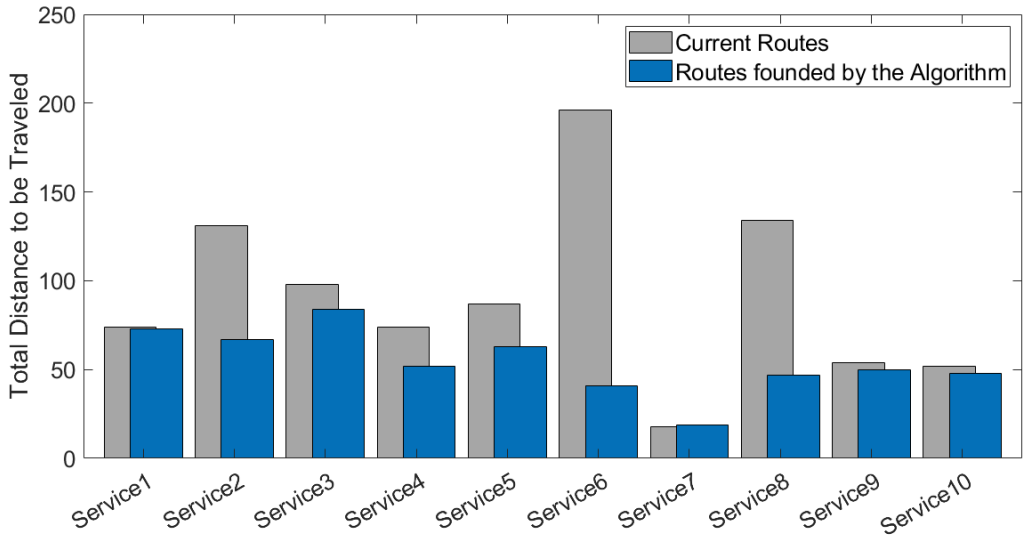


Figure 10. Comparing the total distances.

When the results are examined, it is seen that the routes of all services except *Service7* have been improved with the developed algorithm. The lowest improvement is 1 km for *Service1*, and the biggest improvement is 155 km for *Service6*. When all services are considered, the total distance to be traveled is improved by 40,7% with the proposed algorithm.

6. CONCLUSION

This work considers the Vehicle Routing Problem, which is an well-known combinatorial optimization problem. The Vehicle Routing Problem aims to determine a route plan for a fleet of vehicles sharing uniform capacity, a central (or multiple) depot, and numerous customer requests while striving to minimize the overall route cost, typically measured in terms of the distance covered by the vehicles.

The aim of this study is to find near optimal routes for the services of a private company. Hence, in this project, an algorithm was developed to solve the addresed Vehicle Routing Problem, and a computer program was coded in Python. Then, I applied the developed program on real-life data and compared the results with the current situation. The results show that the total distance to be traveled by the services is improved by 40,7% with the proposed algorithm.

ACKNOWLEDGMENTS

I would like to thank Asist. Prof. Onur UGURLU for his valuable suggestions throughout the project. I would also like to thank company x Denizciler Dokumculuk for sharing service and passenger information with me.

REFERENCES

1. Schneider, M., Stenger, A., & Hof, J. (2015). An adaptive VNS algorithm for vehicle routing problems with intermediate stops. *Or Spectrum*, *37*, 353-387.
2. Mohammed, M. A., Abd Ghani, M. K., Hamed, R. I., Mostafa, S. A., Ahmad, M. S., & Ibrahim, D. A. (2017). Solving vehicle routing problem by using improved genetic algorithm for optimal solution. *Journal of computational science*, *21*, 255-262.
3. Zhang, D., Cai, S., Ye, F., Si, Y. W., & Nguyen, T. T. (2017). A hybrid algorithm for a vehicle routing problem with realistic constraints. *Information Sciences*, *394*, 167-182.
4. Sajid, M., Singh, J., Haidri, R. A., Prasad, M., Varadarajan, V., Kotecha, K., & Garg, D. (2021). A novel algorithm for capacitated vehicle routing problem for smart cities. *Symmetry*, *13*(10), 1923.
5. Huang, S. H., Huang, Y. H., Blazquez, C. A., & Chen, C. Y. (2022). Solving the vehicle routing problem with drone for delivery services using an ant colony optimization algorithm. *Advanced Engineering Informatics*, *51*, 101536.
6. Konstantakopoulos, G. D., Gayialis, S. P., & Kechagias, E. P. (2020). Vehicle routing problem and related algorithms for logistics distribution: A literature review and classification. *Operational research*, 1-30.
7. Reinelt, G. (2003). *The traveling salesman: computational solutions for TSP applications* (Vol. 840). Springer.

8. Korn, G. A., & Korn, T. M. (2000). Appendix B: B9. Plane and spherical trigonometry: Formulas expressed in terms of the haversine function. Mathematical handbook for scientists and engineers: Definitions, theorems, and formulas for reference and review, 892-893.